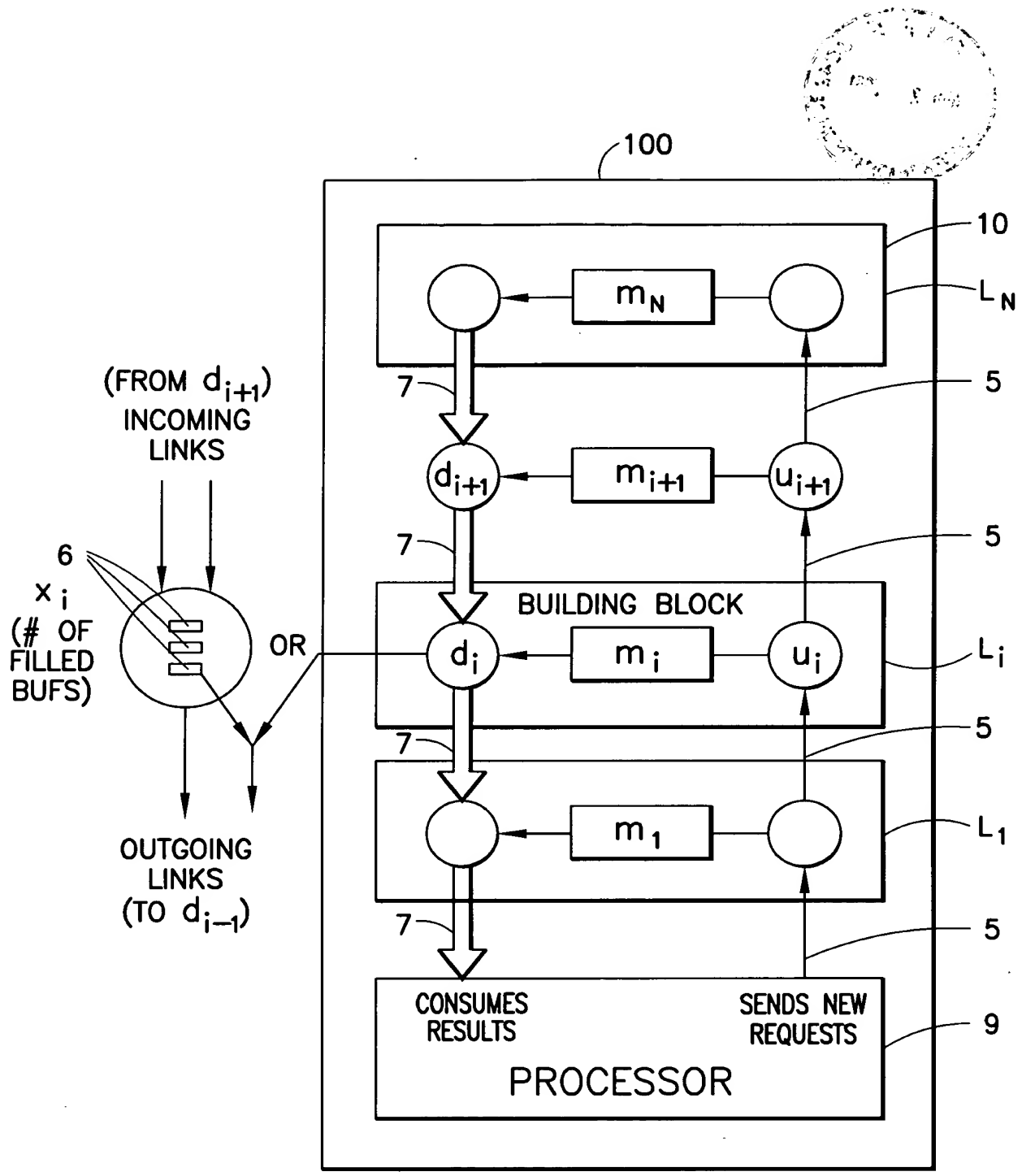
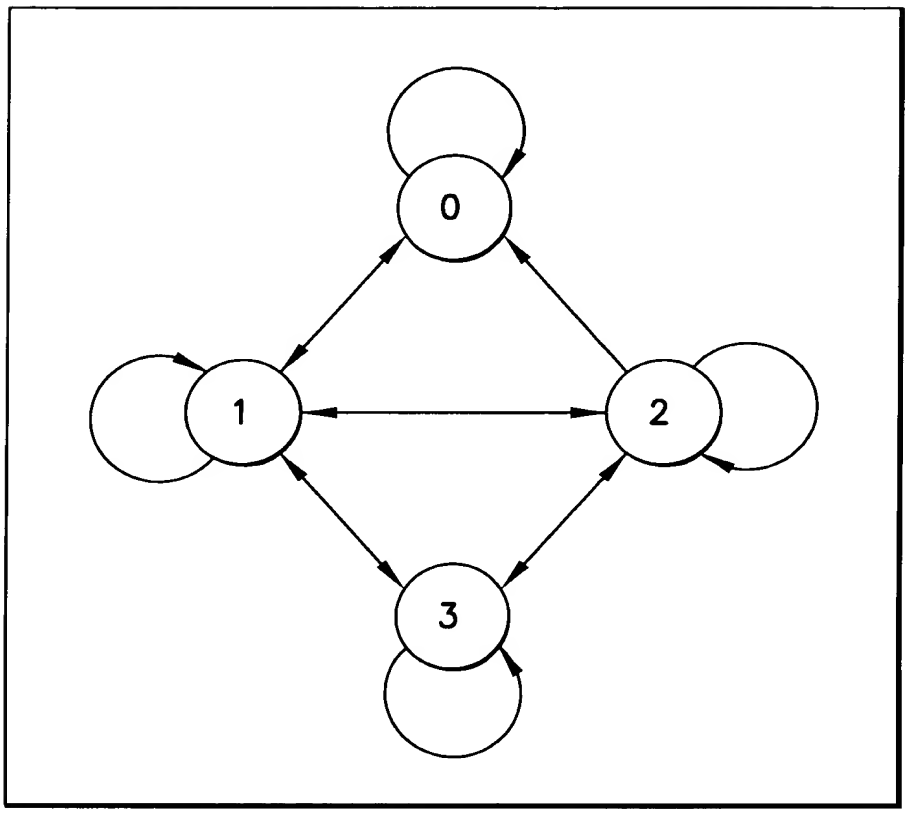

100
10
L_N
m_N
5
7
(FROM d_{i+1})
INCOMING
LINKS
d_{i+1}
m_{i+1}
u_{i+1}
6
x_i
(# OF
FILLED
BUFS)
BUILDING BLOCK
OR
d_i
m_i
u_i
L_i
m_1
L_1
OUTGOING
LINKS
(TO d_{i-1})
CONSUMES
RESULTS
SENDS NEW
REQUESTS
9
PROCESSOR

FIG. 1



INVARIANT:
$(X_i = 3) \Rightarrow$ INCOMING LINKS HAVE AT MOST ONE REQUEST

FIG.2

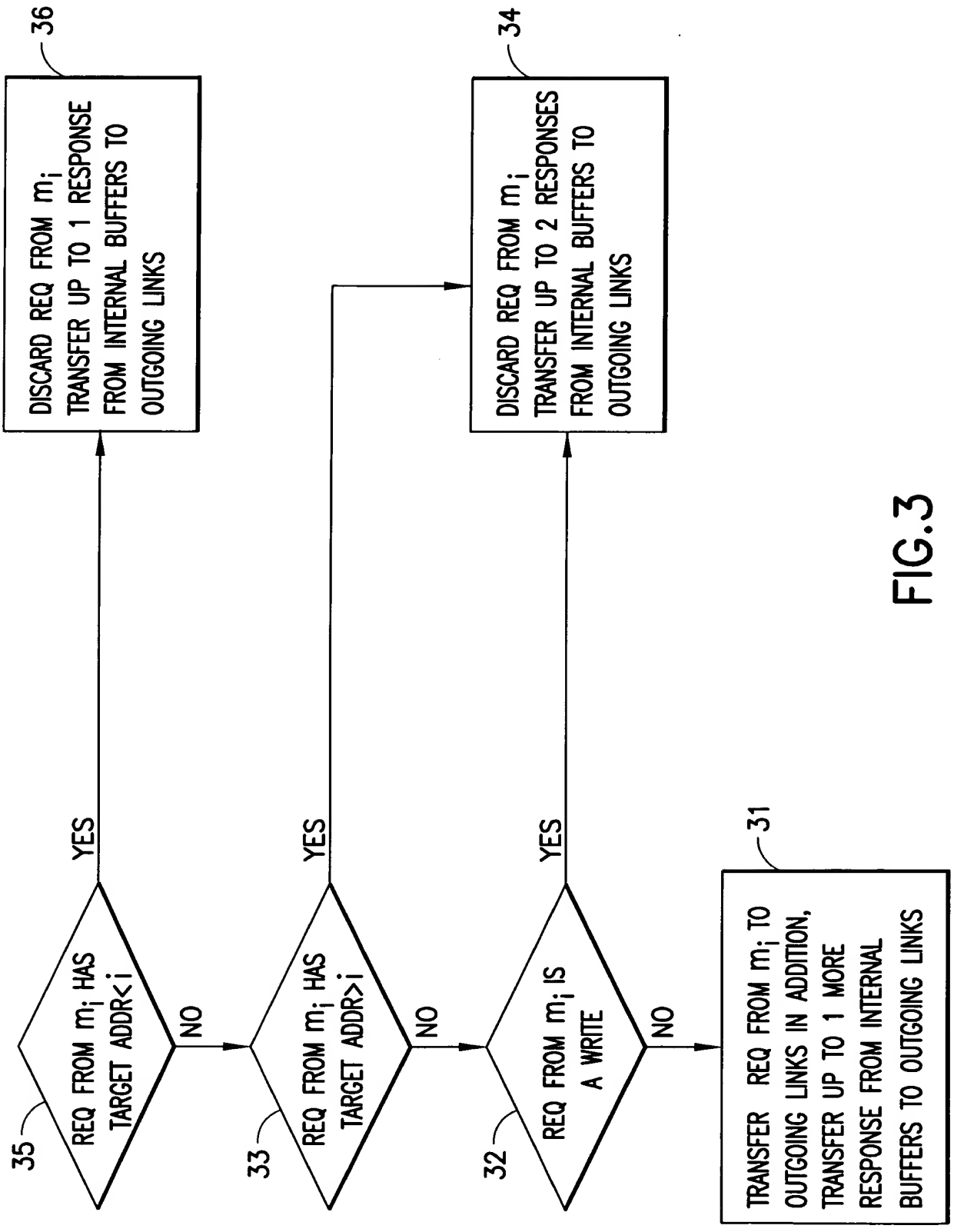

35
REQ FROM m_i HAS TARGET ADDR< i
YES
36
DISCARD REQ FROM m_i; TRANSFER UP TO 1 RESPONSE FROM INTERNAL BUFFERS TO OUTGOING LINKS
NO
33
REQ FROM m_i HAS TARGET ADDR> i
YES
34
DISCARD REQ FROM m_i; TRANSFER UP TO 2 RESPONSES FROM INTERNAL BUFFERS TO OUTGOING LINKS
NO
32
REQ FROM m_i IS A WRITE
YES
31
TRANSFER REQ FROM m_i TO OUTGOING LINKS IN ADDITION, TRANSFER UP TO 1 MORE RESPONSE FROM INTERNAL BUFFERS TO OUTGOING LINKS
NO

FIG.3

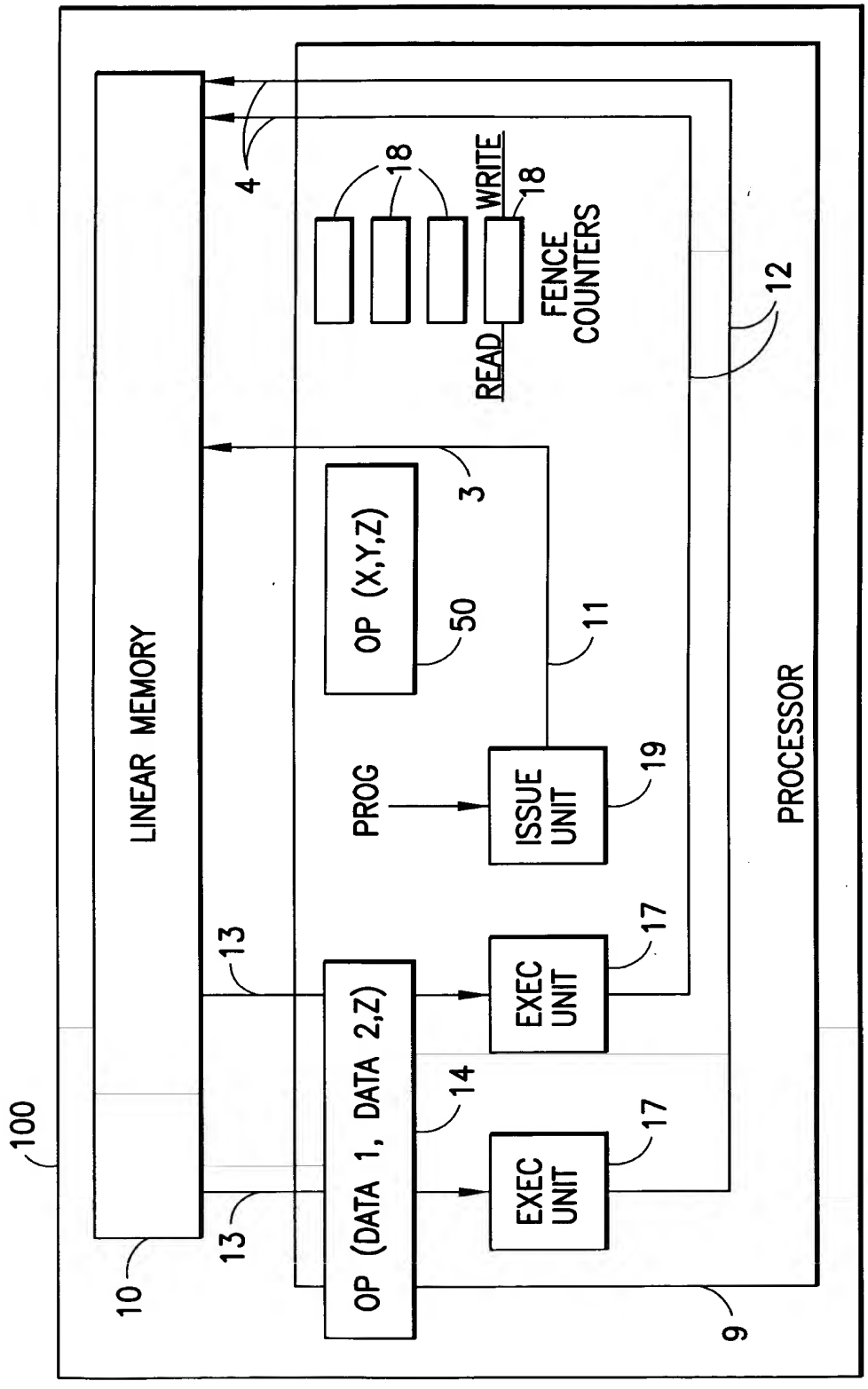

100
LINEAR MEMORY
10
13
13
OP (DATA 1, DATA 2,Z)
14
EXEC UNIT
17
EXEC UNIT
17
PROG
ISSUE UNIT
19
OP (X,Y,Z)
50
11
3
4
18
18
WRITE
READ
FENCE COUNTERS
12
PROCESSOR
9

FIG.4

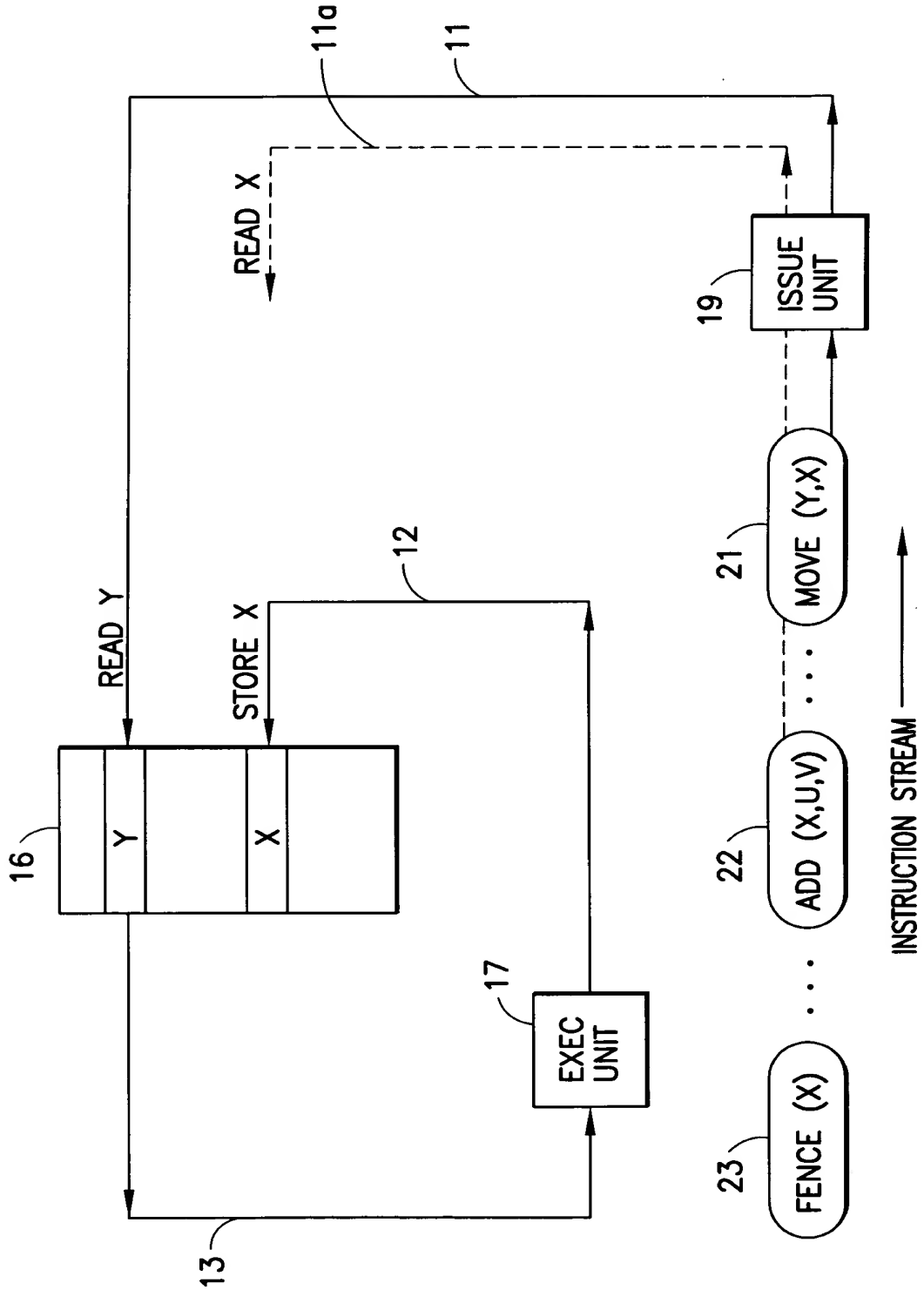

11a
11
READ X
19
ISSUE UNIT
READ Y
STORE X
12
21
MOVE (Y,X)
16
Y
X
22
ADD (X,U,V)
INSTRUCTION STREAM
17
EXEC UNIT
23
FENCE (X)
13

FIG.5

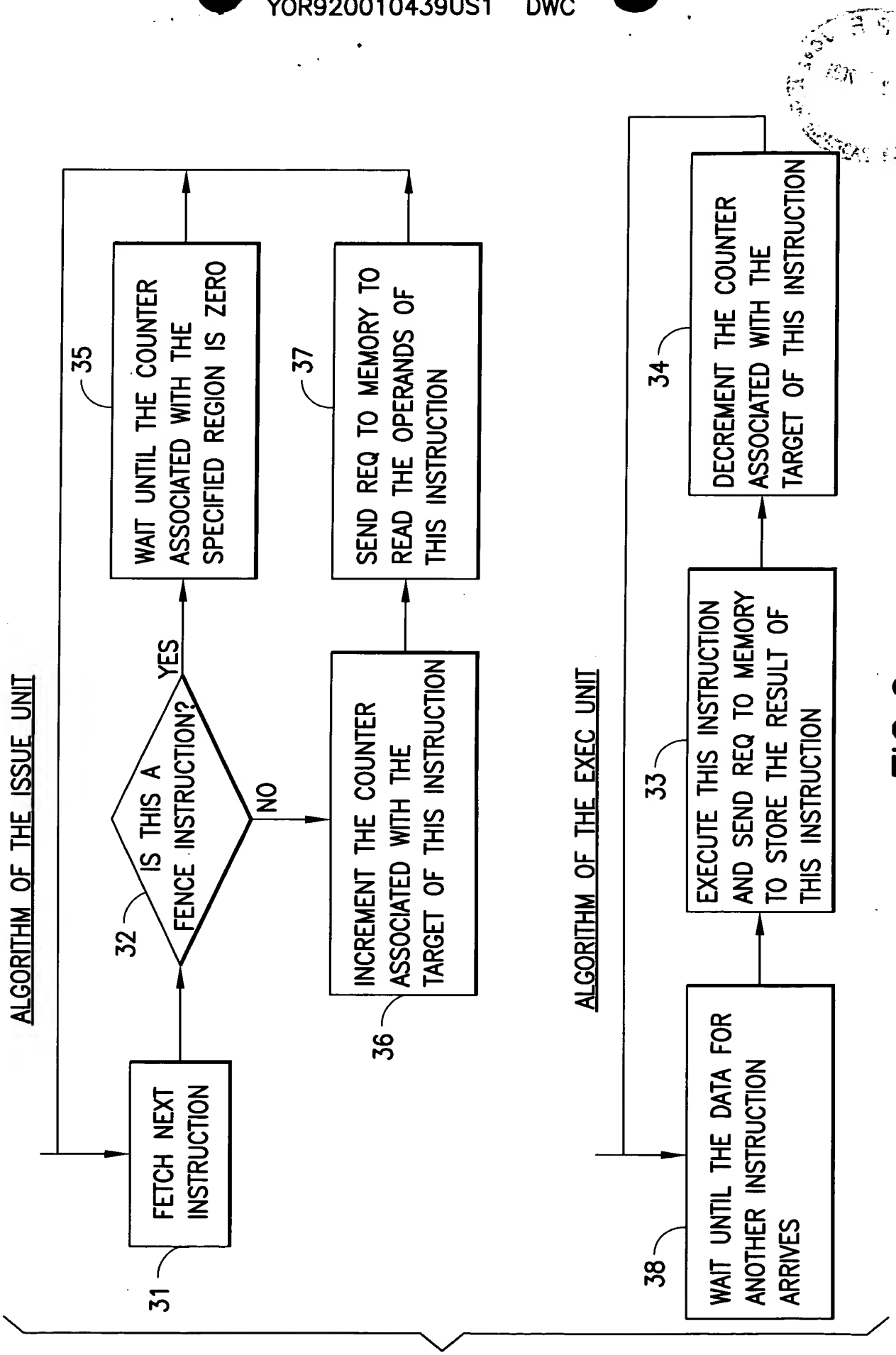

ALGORITHM OF THE ISSUE UNIT
FETCH NEXT INSTRUCTION
31
IS THIS A FENCE INSTRUCTION?
32
YES
NO
WAIT UNTIL THE COUNTER ASSOCIATED WITH THE SPECIFIED REGION IS ZERO
35
INCREMENT THE COUNTER ASSOCIATED WITH THE TARGET OF THIS INSTRUCTION
36
SEND REQ TO MEMORY TO READ THE OPERANDS OF THIS INSTRUCTION
37
ALGORITHM OF THE EXEC UNIT
WAIT UNTIL THE DATA FOR ANOTHER INSTRUCTION ARRIVES
38
EXECUTE THIS INSTRUCTION AND SEND REQ TO MEMORY TO STORE THE RESULT OF THIS INSTRUCTION
33
DECREMENT THE COUNTER ASSOCIATED WITH THE TARGET OF THIS INSTRUCTION
34

FIG.6